CONNEXIONS™

The Interoperability Report

October 1987 Volume 1, No. 6

ConneXions - The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

Introducing Domains.....2

Protocol Testing.....5

Just tell us what you want ...6

Upcoming Events.....8

ISO Seminar Highlights.....9

The 3270 Environment.....10

ConneXions is published by Advanced Computing Environments, 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. Phone: 415-941-3399.

ISSN 0894-5926

From the Editor

In order to solve the problems associated with maintaining a large central name/address database, the Domain Name System is gradually being introduced on the DARPA/NSF Internet. This month we bring you an article by one of its primary architects, Paul Mockapetris of The University of Southern California, Information Science Institute.

As mentioned last month, several things are going on with respect to protocol testing and certification. We asked Bob Jones of Unisys to describe the part his organization is playing in providing a test laboratory for the DoD protocol suite.

Dave Crocker of The Wollongong Group makes a plea for what he calls the Internet Buyers Guide. Such a document would help both the vendors and customers in specifying TCP/IP products.

Our ISO Development Seminar was held last month, and on page 9 we reflect on some of the feedback and lessons learned.

The IBM 3270 family of terminals is perhaps the most widely used data entry device in the world. Because of the particular 3270 architecture, it takes some careful thought to allow these devices to operate over a TCP/IP based network via Telnet. The resulting implementation is known as *tn3270*. One of the major developers of this system is Greg Minshall of UC Berkeley, and in a two-part series starting this month he will describe the 3270 environment and its integration with Telnet.

RFC 1000 was recently issued. As described in our Premiere Issue, this is the RFC Reference Guide for all RFCs from 1 to 999. As well as providing a very useful overview of the RFC collection, it contains a delightful introduction by Steve Crocker on "how it all began". We highly recommend that you obtain RFC 1000 from the Network Information Center. Their number is 1-800-235-3155.

I would like to thank all the people who are contributing articles to *ConneXions*. They provide expert insight into many areas of interoperability. Let me also remind you that your input is most welcome. Let's hear about your experiences and tell us what topics you would like to see covered.

Introducing Domains

by Paul Mockapetris, USC-ISI

Background

In any large internet system, the management of names for hosts, users, etc. becomes a complex task. In the early days of the Internet, this task was easily handled using an ASCII file called HOSTS.TXT, which listed all known hostnames, aliases, and the corresponding host addresses. A site would periodically fetch a copy of this file from the Network Information Center (NIC), which maintained the information in a master copy. It's not too far off to think of a host periodically getting a new copy of this "Internet host phone book" for its users.

Growing host table

However, as the Internet grew, particularly with the addition of thousands of workstations, the "phone book" got very large, and changed very often, even though individual entries were often unchanged for long periods. The cost of FTPing the whole book when only a few entries had changed was getting very high, the NIC was starting to get overloaded with change requests, users who wished to register a new workstation had to wait for the NIC to update the centralized table, and users were beginning to think of new information which could be useful, such as user level entries instead of host entries.

These are the problems that the Domain Name System (DNS) seeks to solve. The DNS changes the way the naming function is performed and adds new capabilities, some of which are already in use, and others which will be defined and phased into use in the future.

The DNS does away with the idea of the monolithic "phone book" and replaces it with a new system of hierarchically organized names called the name space. The name space looks like a tree or org chart; each node in the tree has a short name (called a label) which only has to be different from brother nodes. Because the tree has a single top point, or root, each node has a globally unique domain name which consists of its label and the labels of all of its superiors. The user sees these domain names expressed as strings where dots are used to separate the labels. For example, C.ISI.EDU is a node with label "C", under a node with label "ISI", under a node named "EDU". We use the term "domain" to refer to all nodes below a particular name. Thus ISI.EDU as a domain name identifies a particular node, and ISI.EDU as a domain identifies everything at or below that node.

Domains and Zones

The system used to keep track of these names is also different. An organization can get authority to start a new "phone book" (called a zone) at any point in the tree. For example, ISI obtained permission to manage all of the tree under ISI.EDU, and maintains that zone itself. The system is set up so that ISI could subdivide itself if it wanted to at any point under ISI. Note that this isn't required, but is allowed; the difference between a domain and a zone is that a domain is a complete subtree of the name space, whereas a zone may have internal subtrees removed.

Of course, when your program wants to find out the address of C.ISI.EDU, it has to be able to look in the ISI zone. Rather than getting a copy, it sends off a datagram to a name server known to have a copy of the ISI.EDU zone.

Resolvers

The reply has just the information asked for rather than the whole zone. You don't have to write the code to do this, it is supplied as a part of your operating system's utilities in a program called a resolver. For example, UNIX has a resolver inside the BIND subsystem, and TOPS-20 has one called JEEVES. All of the zones in the domain system are supported by redundant name servers for reliability, and the DNS has a simple mechanism for updating the multiple name servers for a zone.

Cached information

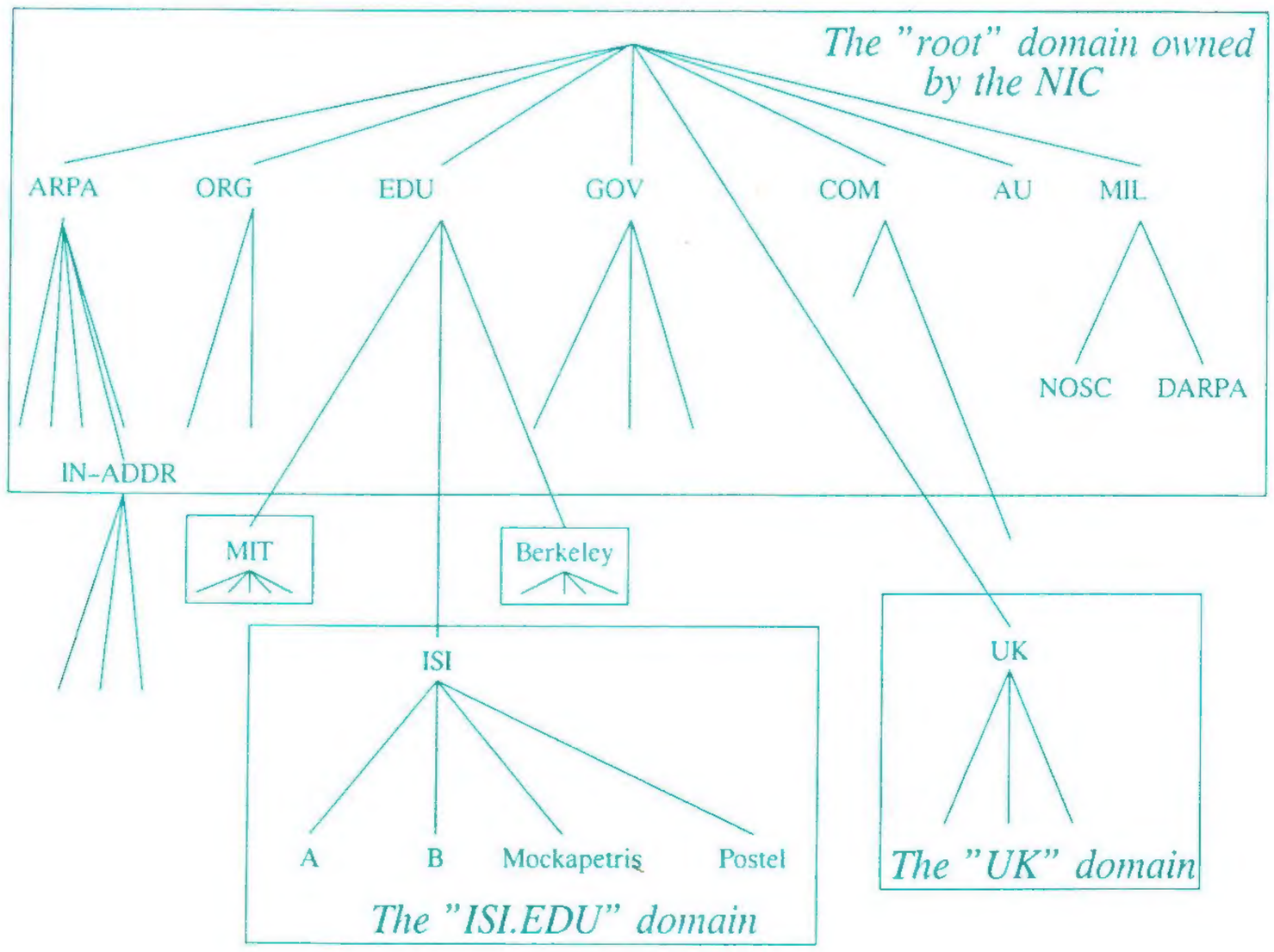
To pursue our phone system analogy, most hosts now call different information operators and write down the results in a cache to avoid repetitive calls to the information service. The cached information comes with a time-to-live (TTL). This time interval that specifies how long the information should be good for. Your resolver discards information with an old TTL.

One cost of all this is that each organization on the Internet will have to organize its part of the namespace. Technically, almost anything goes, but politically, this task is like drawing an organization chart. The top level organization in use today allocates a branch for every country (e.g. US, UK), plus the following functional branches:

MIL - The military (US)
GOV - The other parts of the US government
EDU - Educational institutions
COM - Commercial institutions, e.g. companies
... etc.

The current namespace

The figure below shows a very small part of the namespace; most of the structure and almost all of the lower levels have been omitted. The top levels of the tree are managed by the NIC, which has delegated control of many domains to organizations that manage their own name servers and zones.



continued on next page

Introducing Domains *(continued)*

How does this affect me?

The first thing that happened was that all host names were converted to use the DNS' format, so even if your host still uses HOSTS.TXT, the names have changed to have dots in them. If you are referring to a host in the local domain, the software on your host may help you out by tacking on the local domain name, but if you are referring to a host elsewhere, you may have to type some more characters. For example, here at ISI, I can type "C", and my friendly UNIX knows to try C.ISI.EDU, but if I want to send mail to the NIC, I now have to type SRI-NIC.ARPA instead of SRI-NIC. Of course, if I am using a host at BBN, "C" might mean C.BBN.COM there, but local abbreviation facilities have always been different.

The other thing that happens is that the name lookup procedure is less predictable than it used to be. If you ask to Telnet to some host, the response will be quick if the host's address is already cached, but may take a while if your host has to send off for this information. The delay varies depending on network load and how far away the information is, but can be 10-15 seconds or more. Rarely, no answer comes back. This isn't as bad as it seems since if you can't get to some host's name server, you probably can't get to the host either.

MX Records for mail

Mail is also a little different. If you use full domain names, there shouldn't be any problems, but abbreviated names or old nicknames won't work everywhere, so they should be avoided. Your host software should make most of this painless, but there will be occasional problems until everyone gets the new software running well. On the plus side, the new MX style of mail addressing makes it possible for an organization to route its mail to a set of hosts. This means that you can have backups if a single host is down, and also means that mail-forwarding hosts can represent an organization which isn't directly connected to the Internet. These features should all be buried in software and invisible to the user except that mail addresses should have fewer "%", "!", and other strange parts.

What is in the future?

A lot of this transition has already happened, and more is happening every day. If your host has not converted to the DNS, then things will get less pleasant every day.

The namespace may get reorganized, either locally, or at the top levels. For example, a lot of people feel uncomfortable not having country be the most significant division in the name space. Several places have found that they want to internally divide up their name space differently from what they started with.

The DNS will probably get used for more types of information, such as user mailboxes. This is already happening experimentally in several places. The DNS will also be incorporated as a building block in some new distributed applications, such as NetBIOS.

How do I find out more?

The philosophy and inner workings of the domain system are described in RFCs 882, 883, and 973. RFC 974 describes the changes to the mail system related to domains. RFC 920 describes the current organization of domains. The NAMEDROPPERS mailing list discusses general DNS issues, and specific software, management, applications, and other issues in other mailing lists. The IETF has a working group on domains that coordinates evolution and maintenance of the current system.

Protocol Testing and Certification

by Bob Jones, Unisys

The Defense Communications Agency will soon establish a program for the testing and certification of implementations of the DoD protocol suite. This plan, to be carried out and administered by the National Bureau of Standards, is expected to be announced in the September issue of the Federal Register. The test and certification process will be under NBS's National Voluntary Laboratory Accreditation Program (NVLAP).

Several labs

Under NVLAP, NBS is expected to administer several testing laboratories which will be operated by independent companies. These companies will be reimbursed for their services by vendors offering their implementations for certification. The first laboratory is expected to be open for operation sometime in early 1988.

Automated testing of entire suite

DCA has been involved in the development of a testing capability for the DoD protocol suite (which includes TCP, IP, FTP, Telnet and SMTP) for almost five years. This test system, which is nearing completion and is now under beta-testing, provides a completely automated test generation and reporting environment for the entire suite, including tightly-coupled implementations of TCP/IP. The test system not only tests conformance to the Military Standards, (MIL-STDs) it also tests top-down and bottom-up qualitative functionality. For example, functions that are a part of the protocol definition, but are not under control of the user. These functions are usually found in the basic definition of the service itself, i.e. TCP is defined to provide ordered, reliable datagram service. Furthermore, data from one TCP connection must not "leak" into another connection. The top-down testing cannot validate that duplicate segments are discarded properly, and it cannot guarantee that all the events that might cause a "qualitative failure" occur during testing. However, this testing can determine that bugs associated with normal data transmission and segment-length variations do not occur. Often these anomalies do not occur unless there are relatively intense levels of activity (stress) with a large number of connections operating. Stress testing is also provided.

Other capabilities of the system include testing of protocol stacks, where the interface between application level protocols (FTP, Telnet, and SMTP) and TCP/IP are not well-defined. This particular test is less automated and requires an operator at both the laboratory and at the site under test. Also included is a separate capability to test IP security options.

Testing on the Internet

The automated testing is designed to be carried out across the Internet or via dial-up, directly to a PSN port. Testing across the Internet provides a real-world environment that helps to observe the robustness of an implementation.

The test system to be used for the certification is being developed by the Unisys Corporation under a Research and Development program with DCA's Defense Communications Engineering Center.

Will testing and certification insure total interoperability of protocols on the DDN? No, however it will insure compliance to the MIL-STDs and a level of confidence that interoperability is closer to 100 percent.

Just tell us what you want (A vendor's plea)

by Dave Crocker, The Wollongong Group

What to build and what to buy?

It is truly delightful to see the TCP/IP Protocol Suite migrate into the commercial world. Customers now can obtain stable supported products and vendors can sell new products. This would be a reasonable exchange except for one serious problem: Customers are not quite sure what to tell the vendors to supply. Consequently, vendors are not always certain that they have built the correct products.

Actually, there are two problems. The second one is something that most readers of *ConneXions* have just discovered, because you probably were surprised to hear about the first problem. You thought that the MIL-STD and RFC collection fully described what services are possible and, therefore, what products should be built and bought.

Not true!

Options

While the MIL-STD and RFC documents certainly contain an extensive and sophisticated set of specifications, several factors create some interesting confusion. From the strictly technical side, the specifications are sometimes ambiguous. While this is the most serious cause of multi-vendor interoperability failures (excluding simple implementation errors) it usually occurs early in a product's life-cycle and is remedied fairly quickly through field-testing. More interesting is the fact that some of the specifications, such as Telnet, have *options*. Customers and vendors have limited information about the real need for, or experience with, specific options. In particular, they often do not know how much faith they should have in an option's specification, because it is possible that it has not yet been thoroughly tested.

Standards development cycle

Remember that specifications initially are published as RFCs. "RFC" means "Request For Comments" and a specification does not begin life as a standard. It begins as a *possible* solution to a problem; its publishing is an opportunity for peer review within the research community. If the review is sufficiently favorable, the status of the specification may be elevated to the status of "standard". While the act of elevation may be formal, as with a MIL-STD, it often is not. Adoption may merely be through popularity, creating a *de facto* standard. In either case, this means that *if* you want to do the thing permitted by the specification, then this is the standard way to do it. Over time, the standard may be further elevated to the status of requirement. This means that *all* implementors *must* provide this service, as specified in the standard.

Official Protocols

Unfortunately, there is a very, very limited history with formal statements that indicate when a specification is a requirement. Periodically, an RFC entitled "Official Internet Protocols" is published in an attempt to remedy just this thing. (The current number is RFC 1011). While it is an excellent beginning, you, the customer, and we, the vendors, need more.

Unpublicized knowledge

First, this RFC itself needs some review, and second, its granularity of reference is too large. That is, it indicates the status of the entire specifications, with no information about *portions* that may not be required.

And this touches the non-technical side of the problem: The TCP/IP suite of protocols, as with any sophisticated system, has developed a *cultural milieu*. There is a style of use and, frequently, an informal and unpublicized community of understanding about the features that *really* are required and the ones that are not.

For example, are all the File Transfer Protocol (FTP) commands supported by a typical implementation? Well, as was observed in the previous issue of *ConneXions*, the PASV command is not universally provided. How do you find out other such tidbits about FTP? What about other protocols?

Currently, you must conduct extensive discussions with the DDN Internet Engineers, as well as having to experiment with many implementations. This is expensive, slow and error prone, even if you are lucky enough to coerce one or more of the Internet Gurus into a question-and-answer session.

But perhaps you are not yet convinced of the problem. Perhaps a little story will help:

A story

Once upon a time, in a corporate engineering lab, far, far away, a Computing Facilities manager was tasked with writing a Procurement Specification so that her lab could use TCP/IP. Well, our lonely manager sat down and read all the specs and she remembered and understood more of them than had anyone else who had read them for the first time. (You can tell this is a fairy tale -- How many customers can afford to have someone put in that much effort, to buy a single product?)

Telnet, a symmetric protocol

What sort of thing does our herione read? Let's take a look at Telnet. She discovers that Telnet is a symmetric protocol, so that functions can apply equally to both *client* programs and *server* programs. Further, she reads that a connection is full duplex and that the WILL/WONT, DO/DONT negotiation mechanism controls the invocation of Telnet Options. Lastly, she notes that the Official Internet Protocols RFC says that a purchaser of Telnet code simply cannot live without the Remote Echo option in client and server Telnets and each side must support both the request (issuing a DO) and the acceptance (issuing a WILL) modes.

Those of you among the Telnet cognocenti may already have discerned the problem: Having a client Telnet able to issue a WILL Remote Echo means that data *from* the server (i.e., the remote machine that the user logs into) will be echoed *back* to it! While legal, this is a Bad Thing.

So what was our manager's error? Very simply, she didn't know that Remote Echo is only intended for having the server perform echo (WILL ECHO) and that no one has client servers issue a WILL. But then, this fact is not recorded anywhere.

continued on next page

Just tell us what you want *(continued)*

By the way, this story really happened and variations of it happen all the time.

Fortunately there is a relatively straightforward solution to this problem that also results in the saving of effort. On the other hand, it moves the effort from one group of people to another.

Acquisition documents, such as government Request for Proposals, (RFPs) usually contain a set of lists, with each list stating the specific features that must be provided for a given protocol or service. Currently, the writer of each RFP must generate the list, although copying from other RFPs is not uncommon.

So what *am* I proposing can be done to address this issue?

Shopping list needed

I hereby plead for the Internet Activities Board to commision the development of a single approved version of these lists, possibly called Requirements for Standard Implementations of the DoD Protocol Suite.

The existing MIL-STD and RFC protocol specifications are needed by implementors. Think of this new document, with concise feature lists, as being The Internet Buyer's Guide.

DAVE CROCKER is Vice President of Software Engineering at The Wollongong Group, a provider of TCP/IP implementations on a variety of hardware platforms. Before migrating himself into the commercial world, Dave was a participant in the ARPANET research community for ten years, mostly in the arena of electronic mail. He is editor of RFC 822, "Standard for the Format of ARPA Internet Text Messages" and was principal contributor to the development of MCI Mail.

Upcoming Events

Advanced Computing Environments is conducting a survey to determine "what works with what" in the TCP/IP world of computer networking. We call it *The TCP/IP Interoperability Survey*. Many of you will already have received copies of our survey form. If not, please feel free to contact Ole Jacobsen at 408-996-2042 to receive more information and participate in the survey.

Our next conference is the *2nd TCP/IP Interoperability Conference* which will be held December 1-4, 1987 at the Hyatt Regency Crystal City in Arlington, VA. A special *User/Vendor Forum* session will take place on Friday, December 4. There are sometimes intricate problems that arise in operating a multivendor network, and users should use this session as a place to get problems aired and resolved. Contact Advanced Computing Environments with your concerns, questions and problems to ensure that they are included in the agenda. For complete details, including conference program and registration, contact us at 408-996-2042.

ISO Seminar Highlights

Some 250 people attended the ISO Development Seminar which was held in Monterey from August 31st through September 2nd 1987. Nancy Hall, Sue Lebeck, Rob Hagens, Marvin Solomon, and Lawrence Landweber from the University of Wisconsin at Madison described their efforts in implementing the Network, Transport, Session, Presentation, and Message Handling parts of the ISO stack. They were joined by Marshall Rose from Northrop Research and Technology Center who described his FTAM implementation.

The breakdown of attendees was:

44% Vendor
24% User
20% System Integrator
12% Military

The three day seminar gave the audience an early opportunity to fully explore the implementation experiences of a set of R&D developers of emerging ISO standards. In addition to presentations by each of the speakers, a question-and-answer session was held which provided useful information exchange.

What was learned?

- The ISO/OSI set of standards represent a very large and ambitious undertaking. Early developers face a myriad of choices if they are to ever get a product built.

- ISO standards definitions are slightly buggy and a moving target. The bugs aren't devastating, but the moving targets cause some concern.

- Some important standards are still incomplete or missing: Directory Services, Intermediate System to Intermediate System routing, Virtual Terminal, Connection-oriented to/from Connection-less transport protocol interoperability, Network Mangement, Security.

- Users really want ISO. They believe in it's benefits. They are concerned about the (possible) simultaneous availability of numerous products so that actual usage can be made.

- The audience wanted to hear more from the actual architects of ISO protocols. They want to hear the schedule and the plans for filling in the blanks. It clearly takes great coordination to make it all work.

- TCP/IP had it alot easier! It had a chance to get developed and fleshed out in laboratories for many years before it got into widespread use. In contrast, the world expects ISO to spring out full born on some magic day.

tn3270 Part One: The 3270 Environment

by Greg Minshall, UC Berkeley

What is an IBM 3270 terminal?

In the specific, an IBM 3270 is a piece of hardware which contains a display head and a keyboard, and allows a host program to interact with a user sitting at the IBM 3270 (it is also possible that the IBM 3270 is actually a printer, in which case there is minimal human input).

IBM 3270 terminals can have as few as 12 lines, and as many as 43. They come in models with either 80 columns or 132 columns. There are two-color ("black and white") models, four-color models, and models displaying more than four colors.

Controllers and terminals

Each terminal is connected to a terminal controller (current controllers include the IBM 3274 and IBM 3174). The controller is, in turn, attached to the IBM host system (through a channel connection, or via a synchronous [SDLC or BSC] serial line). In the earliest 3270's (known as 3277's), and in the most advanced styles of 3270's (known as Distributed Function Terminals), the terminal contained most of the logic related to processing the 3270 data stream; in other styles of 3270's (known as Control Unit Terminals), the terminal itself has little logic beyond that needed to display characters on a screen and to pass individual keystrokes to the terminal's controller. The reason for this migration of function has been a result of some rather obscure marketing strategy, plus economies of silicon chips, and has no effect on either the user or programmer (aside for some timing differences); both of these individuals see the controller/terminal combination as a single entity. Henceforth in this article, we will use the term terminal even when controller might (for some implementations) be the correct term.

The 3270 protocol

In the abstract, an IBM 3270 is defined by a protocol between a terminal and a host. The bytes and structures passed back and forth between terminal and host are known as the 3270 data stream. There are various rules on which bytes/structures are valid and when. Sometimes the rules depend on the characteristics of the specific hardware implementation of IBM 3270. For example, when addressing an IBM 3270 which has only 24 lines, an attempt to address line 32 would be in error (though many IBM terminals will ignore this particular error condition).

The terminals are known as synchronous terminals. All communication between the host and the terminal is controlled by the host. The major exception to this is that some classes of terminals can indicate a request for service to the host in an asynchronous fashion (this request for service comes in the form of an attention interrupt from terminals connected to channel attached controllers).

One difficulty in understanding what really constitutes an IBM 3270 is the way the IBM 3270 protocol is intermixed with the underlying transmission protocol. IBM 3270's which are connected via IBM's Systems Network Architecture (SNA) behave in ways different than those connected via a non-SNA path.

For example, in an SNA connected 3270, there is the concept of a presentation space owner, ie: who is allowed to modify the screen at any given time. The owner is either the keyboard (and, thus, the user typing at the keyboard), or the host application program. There are rules for switching the ownership of the screen back and forth, and error conditions if the wrong entity attempts to modify the screen. This concept is absent from, for example, channel attached, non-SNA, 3270's. In this article, we do not claim to have made the correct splitting of functions into the various layers, since, among other reasons, the term "correct" is subject to religious debate.

Half-duplex dialog

One important design assumption of the IBM 3270 terminal family is that at any given point in time either the user of the terminal will be typing at the terminal, or the host will be displaying information on the screen or processing previous user input. So, the implementations automatically lock the keyboard whenever the user has "finished talking" to the host (see below).

Field oriented screen

The 3270 screen is field oriented. The host displays a formatted screen, and the user (using the keyboard) fills in various input fields. After filling in the information, the user indicates to the terminal (via the Enter, Clear, or Program Function keys) to transmit (portions of) the screen to the host. At this point, the keyboard locks; the host reads the screen, processes the user's input, and displays a new screen, based on the previous screen and the user's input; and finally the host unlocks the keyboard.

Each field on the screen is delimited by an attribute byte (which displays as a blank). Each attribute byte defines the display and functional characteristics of the field. Among the characteristics that an attribute byte can define are: highlighting (via color, high intensity, underlining, etc.), visibility, output only, selector pen detectable, numeric.

EBCDIC

The IBM 3270 keyboard contains basic text keys (alphabet, numbers, and special characters). Most (but not all) 3270's converse with the host in IBM's EBCDIC (Extended Binary Coded Decimal Interchange Code), an 8-bit code which is roughly equivalent to 7-bit ASCII. Unfortunately, there is no unique mapping of ASCII into EBCDIC, nor of EBCDIC into ASCII. The graphic representation space of neither code is a subset of the other. The 3270 itself is capable of displaying graphics from both the EBCDIC and ASCII code sets; however, access to these graphics is sometimes difficult to achieve within the 3270 data stream.

Special function keys

In addition to text keys, the 3270 keyboard contains a number of local editing keys. These keys typically affect either one character, or a group of characters delimited by a field or fields. Examples of these keys are: delete, field tab, erase to end of field, erase input, and field back tab. There are keys to move the cursor up, down, left, and right (as well as to the home position). A user can invoke (and later reset) insert mode.

continued on next page

The 3270 Environment *(continued)*

The final group of keys are those which cause interaction with the host to occur. These are known as the AID generating keys. AID stands for Attention Identifier, and is a code sent to the host to represent the specific AID generating key pressed by the user.

One of the AID generating keys (clear) affects the screen as well as initiating communications with the host; the rest of the AID generating keys simply lock the keyboard and inform the host of the user's action.

By design, most user interaction takes place between user and terminal, not between user and host. Thus, entry of data into the input fields on the screen, the editing of this data, moving the cursor around, etc., all occur without communication to the host. When there is communication with the host (eg: when an AID generating key is pressed), then only those fields on the screen which have changed "recently" are sent to the host.

Reading and writing from/to the screen

The inbound data stream (terminal to host) generally consists of an AID, followed by information about where the cursor was at the time the data was sent inbound, and also the contents of those fields recently modified. The outbound data stream (host to terminal) consists of a command code (read, write, etc.). In the case of a write command (and there are various flavors), the command code is followed by a write control character (WCC; the WCC, among other functions, can cause the 3270 alarm [bell] to sound, and can unlock the keyboard [if the host feels the need of some user input]) and then by a sequence of bytes which consist of data to appear on the screen and orders. Orders contain information needed to begin new fields, modify old fields, set portions of the screen to a constant value, move the cursor, etc.

3270 and graphics

In addition to serving as a simple text entry/display terminal, IBM has extended the 3270 protocol to do graphics. One early scheme (still in use many places today) involved attaching a Tektronix 4010 display head to the backend of the earliest IBM 3270 terminal (known as the 3277). This addition to the protocol defined commands for addressing the 4010, and even allowed for single character input from the 3277.

More recent, but still traditional, IBM graphics products make use of Programmed Symbols. Programmed symbols basically give the programmer a way of defining a character set font (of fixed width). There are some uses for this facility, but in general it is difficult to do graphics using programmed symbols.

The newest IBM 3270 terminals support bitmap (APA) and vector graphics interfaces (in addition to some programmed symbol support). These terminals are new enough that not very many people have experience working with them. They do, however, provide a model of graphics which most programmers feel more comfortable with.

Printers

As mentioned briefly above, the IBM 3270 family contains printers as well as terminals. The most traditional way of integrating printers has been to connect a printer to the 3270 controller (in the same way the 3270 display/keyboard is connected to the 3270 controller).

This allowed the host to access the printer by sending 3270 data streams to the printer (3270 data streams formatted, of course, for consumption by a printer rather than by a 3270 display). Additionally, 3270 display stations were permitted shared access to the printer by one of two methods.

In the first method, the user would request help from the host to print the 3270 screen on a certain printer. The host would cause the screen image to be printed on the printer. The second method bypassed the host entirely by allowing the site administration to designate, for each display station, a specific printer which screen print requests would be sent to. In this method, the printer sometimes appears to become busy asynchronously (as seen by the host or by a display station user) as the printer services first one then another of its "clients".

There are plain text printers, color printers, and graphics printers available.

Some of the newer IBM 3270 terminals have the ability to attach a printer directly to the terminal. One class of terminal in which this ability is present is that consisting of IBM Personal Computers with some mix of 3270-emulating hardware and software.

3270 and the IBM PC

The onslaught of IBM PC's in the workplace has had its effect on the IBM 3270 protocol and what people are doing with that protocol. Initially, there was a high demand for 3270 emulation within a PC solely to reduce two boxes (one a PC, one a 3270) on various workers' desks to one box capable of performing both functions. In this mode, one was either in PC-DOS mode, or in 3270 mode, and there was little interaction between the two modes.

It was soon recognized, however, that the 3270 connection (potentially connected via a local channel and coax cable with an end-to-end transmission speed of about 50 kbytes/second) offered the ability of programmatic access between the PC and the host. Thus, the various boards and systems which provided 3270 emulation from a PC began to provide an Application Programming Interface (API) allowing PC programmers to inspect and modify the 3270 screen image and status, and to cause data to flow to and from the host system. This ability is used, for example, to automate logon's to the host system; to transfer files to and from the host system; and to implement various utilities such as host-based file stores, access to the host printer, etc.

The one deficiency of the API is that the PC programs using the API are very complicated. They need to handle various conditions which may unexpectedly change the screen contents (such as warning messages from operators, etc.). They need to determine exactly when, and where, they can enter data on the screen.

continued on next page

The 3270 Environment *(continued)*

If they are attempting to transfer data, then the programs need to provide flow control, sequence numbering, reliable delivery, etc., of data. The end result is that the PC application programmer ends up spending an inordinate amount of time worrying about details which have little to do with the specific application under development.

In June, 1986, IBM announced a new facility for programmatic access between 3270-attached PC's and hosts. This new facility is known as the Server Requester Programming Interface (SRPI). SRPI is, basically, a remote procedure call mechanism. A PC program, using SRPI, can specify the name of a function to be invoked on the host and pass certain parameters (and data) to that function.

SRPI, which runs both in the PC and within the host operating system environment, then transfers the request (reliably) to the host. The host function (either one provided by IBM, or one written by a host programmer) is then invoked to process the request. After processing the request, the host program notifies SRPI of the result of the function (which may involve changes to the value of some parameters, a return code, and some data), which results are finally communicated back to the requesting PC program.

The SRPI facility is clearly important for the future growth of the PC programmatic access to host functions.

Telnet and IBM 3270

Within the TCP/IP protocol suite, there are three major application layer protocols. One protocol (FTP - file transfer protocol) moves files over the network; one protocol (SMTP - Simple Mail Transfer Protocol) moves mail over the network; the third (Telnet) allows for terminal connection over the network.

Network Virtual Terminal

Telnet was designed as a protocol which would be general enough to allow any terminal connected to a computer with a Telnet client to access any host system possessing a Telnet server (assuming, of course, the two machines are connected over a network). To achieve this generality, the specification of Telnet necessarily lies in the intersection of the capabilities of "all" terminals. This intersection allows for the entry and display of characters of the ASCII code set. The intersection (known as the Network Virtual Terminal, NVT) basically provides for a teletype-like interface between terminal and remote host.

The IBM 3270 can operate in NVT mode (with some restrictions). The NVT model consists of lines appearing on a display screen. Since the 3270 has a display screen, and since the local Telnet client program can manipulate the screen, output from the server host to the terminal can be displayed (assuming the client Telnet provides an ASCII-EBCDIC translation function). For input, the 3270 keyboard provides the mechanism for the user to format a line to be transmitted over the Telnet connection to the server host. However, there is no equivalent, within the NVT model, of a clear key, a program function key, or any other such key. So, a Telnet client program will likely reserve the use of such keys to its own use (ie: typing program function 10 might mean "enter Telnet command mode").

Line-at-a-time

Additionally, there is no way that the Telnet client program can achieve cognizance of each character typed by the 3270 user; thus, only complete lines (with or without a trailing carriage return-line feed) are likely to be transmitted over the network.

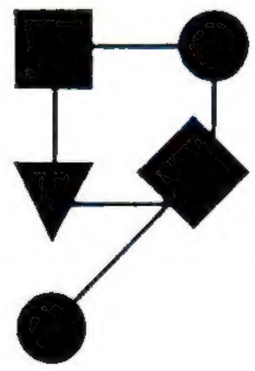
Telnet Options

In order to allow terminals and server hosts to communicate in a more natural way, Telnet contains sub-protocols (known as Telnet options) which allow for extending the protocol between mutually consenting entities. For example, the client can (if the server asks) inform the server as to the specific terminal type in use by the end user by using the terminal type option. The two ends can agree to leave the intersection code set (know as Network ASCII) by agreeing to the binary option.

These extensions, for example, can be used to allow terminals capable of advanced graphics or text manipulation the ability to communicate in native mode with programs on the remote (server) host (assuming these server programs know the specific protocol used by the specific terminal). In fact, since many ASCII terminals do not use any code points outside the Network ASCII code set, many specific terminals can be driven in a native mode without ever leaving the NVT-mode of operation. (One important class of terminal which is not able to use NVT-mode is that class which assigns semantics to the null [0] character, since that character is quite often deleted by the Telnet client program. Certain Tektronix terminals fall into this category.)

Next month we will further examine how a 3270 device can operate over Telnet. Part Two is entitled "IBM 3270 Data Stream over Telnet".

GREG MINSHALL is a programmer at the University of California at Berkeley.



CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040

FIRST CLASS MAIL
U.S. POSTAGE
P A I D
SAN JOSE, CA
PERMIT NO. 1

CONNEXIONS

PUBLISHER Daniel C. Lynch

EDITOR Ole J. Jacobsen